Matrices, Cont’d

Storing two-dimensional numerical data

---

Analyzing table data

<table>
<thead>
<tr>
<th>level</th>
<th>1998</th>
<th>1999</th>
<th>2000</th>
<th>2001</th>
<th>2002</th>
<th>2003</th>
<th>2004</th>
<th>2005</th>
</tr>
</thead>
<tbody>
<tr>
<td>advanced</td>
<td>7</td>
<td>9</td>
<td>15</td>
<td>18</td>
<td>20</td>
<td>24</td>
<td>29</td>
<td>35</td>
</tr>
<tr>
<td>proficient</td>
<td>17</td>
<td>15</td>
<td>18</td>
<td>27</td>
<td>24</td>
<td>27</td>
<td>28</td>
<td>27</td>
</tr>
<tr>
<td>needs improvement</td>
<td>24</td>
<td>23</td>
<td>22</td>
<td>30</td>
<td>31</td>
<td>29</td>
<td>28</td>
<td>24</td>
</tr>
<tr>
<td>failing</td>
<td>52</td>
<td>53</td>
<td>45</td>
<td>25</td>
<td>25</td>
<td>20</td>
<td>15</td>
<td>15</td>
</tr>
</tbody>
</table>

Statewide results for MCAS Test in Mathematics, Grade 10
Indexing with colon notation

To refer to an *entire column* of a matrix, provide : as the first index and the column number as the second index

```
>> nums(:, 3)
ans =
3
8
13
18
```

To refer to an *entire row* of a matrix, provide : as the second index and the row number as the first index

```
>> nums(2, :)
ans =
6 7 8 9 10
```

Plotting trends in performance levels

We begin our analysis by plotting the data for each performance level over the 8 years

```matlab
% create matrices that store data and years
results = [ 7 9 15 18 20 24 29 35; ... 17 15 18 27 24 27 28 27; ... 24 23 22 30 31 29 28 24; ... 52 53 45 25 25 20 15 15];
years = [1998 1999 2000 2001 2002 2003 2004 2005];
```

Each row of the table corresponds to a performance level. How do we plot the resulting trend over the given years?
Plotting the data

```matlab
hold on
plot(years, results(1,:), 'b', 'LineWidth', 2);
plot(years, results(2,:), 'g', 'LineWidth', 2);
plot(years, results(3,:), 'c', 'LineWidth', 2);
plot(years, results(4,:), 'r', 'LineWidth', 2);
hold off

xlabel('year')
ylabel('percentage of students')
title('MCAS results')
legend('advanced', 'proficient', 'improve', 'failing');
```

Finally, ...

Suppose we want to print the change in results between 1998 and 2005 for each performance level...

How do we do this?

Change in performance between 1998 and 2005:
advanced: 28%
proficient: 10%
needs improvement: 0%
failing: -37%
Printing changes in results

% print total change in results between 1998 and 2005

```matlab
totalChange = results(:, end) - results(:, 1);
disp('Change in performance between 1998 and 2005:');
disp(['advanced: ' num2str(totalChange(1)) '']);
disp(['proficient: ' num2str(totalChange(2)) '']);
disp(['needs improvement: ' num2str(totalChange(3)) '']);
disp(['failing: ' num2str(totalChange(4)) '']);
```

Change in performance between 1998 and 2005:
advanced: 28%
proficient: 10%
needs improvement: 0%
failing: -37%

Time-out exercise

For each year, compute a **weighted sum** of the four percentages, using a weight of 1 for “advanced”, 2 for “proficient”, 3 for “needs improvement” and 4 for “failing”*

```matlab
overallPerformance =
```

Add a new row to the **results** matrix that stores these weighted sums

* The resulting sum can range from 100 (great!) to 400 (not so good...)
More indexing with colon notation

We can use colon notation to refer to a range of indices within a column or row of a matrix.

```matlab
>> nums(1:3, 4)
an =
   4
   9
  14
>> nums(3, 3:5)
an =
  13  14  15
>> nums(2:3, 2:4)
an =
   7   8   9
  12  13  14
```

Conditional operations on matrices

A conditional expression can be applied to an entire matrix all at once producing a new matrix of the same size that contains logical values.

```matlab
ages = [13 52 19 21; 18 47 23 15; 60 38 16 12];
teens = (ages >= 13) & (ages <= 19);
```

```
ages
13  52  19  21
18  47  23  15
60  38  16  12
```
Using logical vectors

```matlab
>> ages(teens) = 0
ages =
    0 52  0 21
    0 47 23  0
   60 38  0 12

>> overTheHill = ages(ages>40)
overTheHill =
   60
   52
   47
   18
   47
  23
   15
```

Tip on min & max

```matlab
>> nums1 = [4 1 7];
>> [minVal minInd] = min(nums1)
minVal =
    1
minInd =
    2
>> [maxVal maxInd] = max(nums1)
maxVal =
    7
maxInd =
    3
```

What if there's more than one occurrence of the min or max value?

```matlab
>> nums1 = [4 1 7 1];
>> [minVal minInd] = min(nums1)
minVal =
    1
minInd =
    2
```

How could you create a logical vector that's true wherever the min value appears?
**Time-out exercise**

Given:
- matrix `medals` with three rows that store number of gold, silver & bronze medals won by countries in the 2018 Winter Olympics
- cell array of corresponding country names
- vector of populations for each country

```matlab
medals = [9 0 6 13 4 1 12; ... 5 5 5 11 3 0 7; ... 7 2 6 9 2 3 5];
names = {'Canada' 'China' 'US' 'Norway' 'South Korea'... 'Great Britain' 'Germany'};
pops = [36624200 1409517400 324459500 523300 ... 50982200 65640000 82114200];
```

**Write code to do the following:**

1. create a vector with the total medals for each country
2. print the name of the country with the most total medals
3. print the name of the country with the largest per capita total medals
4. print the names of countries with more bronze medals than gold and silver medals combined
5. print the names of countries with the least number of bronze medals